Implementación del control de un atenuador programable de RF mediante EDU-CIAA

Camino, Lucas Agustín
Facultad de Ingeniería UNLP
La Plata, Buenos Aires, Argentina
lucascamino95@gmail.com

Figueroa, Lihuen León
Facultad de Ingeniería UNLP
La Plata, Buenos Aires, Argentina
lihuenfigueroa@gmail.com

Corrao, Santiago, Vicente
Facultad de Ingeniería UNLP
La Plata, Buenos Aires, Argentina
santiagocorrao96@gmail.com

Sager, Gerardo Enrique
GridComD Facultad de Ingeniería UNLP
La Plata, Buenos Aires, Argentina
ger@ing.unlp.edu.ar

***Resumen*—El presente documento describe la implementación de un sistema de control basado en la EDU-CIAA para un atenuador programable. El mismo formará parte de un sistema de mediciones de RF que permitirá la calibración y ensayo de transmisores destinados a sistemas de recolección de datos satelitales con frecuencias de hasta 2GHz.**

***Palabras clave*— instrumentación, atenuador programable, EDU-CIAA**

I. Introducción

A partir de la necesidad de implementar un sistema de mediciones destinado a calibrar y ensayar equipos transmisores de RF, el cual necesita un atenuador variable, y la existencia en el Laboratorio de uno de ellos de marca Aeroflex modelo 3200-1, que permite variar la atenuación entre 0 y 127 dB, se decidió implementar el control del mismo mediante una EDU-CIAA complementada con una placa mezzanina habitualmente llamada "poncho" [1] por sus usuarios. Dentro de la línea de productos que ofrece Aeroflex, existen modelos con interfaz TTL, sin embargo, el modelo del que se dispone necesita niveles de tensión de 12V para su comando. Se decidió que el diseño del "poncho", incluyera el hardware necesario para accionar el atenuador, a partir de los niveles TTL provistos por la EDU-CIAA, mediante el uso de opto acopladores lo que significa una mejora en la confiabilidad y robustez del instrumento.

II. Objetivo

El objetivo de este trabajo es implementar un sistema destinado a controlar la operación del atenuador programable. El control estará comandado manualmente mediante un teclado y un display o bien remotamente a través de interfaces USB, RS232 y/o RS485. Esto permitirá acoplar el conjunto a un sistema de mediciones automatizado.

III. Diseño de hardware

A. *Selección del microcontrolador*[2]

El controlador principal del sistema fue la placa EDU-CIAA NXP. La elección se debió a factores como la robustez de la placa, la diversidad de interfaces serie que posee (se utilizaron la USB, la RS232 y la RS485), así como los numerosos puertos de propósito general. Asimismo se simplifica el diseño de hardware al estar prevista la interfaz con una placa de circuito impreso adicional, habitualmente denominada "poncho" por los usuarios de la EDU-CIAA. No se puede dejar de destacar la existencia de abundante documentación en cuanto al hardware[3] y al software del proyecto CIAA.

B. *Diseño del "Poncho"*

Fueron necesarios el diseño y la construcción de un poncho específico para la conexión y protección de los distintos componentes y elementos, así como para el acondicionamiento de las señales de tensión (Fig. 1).

Un requerimiento importante del proyecto es la tensión de 12V necesaria para accionar el atenuador, en efecto, al tratarse de un componente costoso y delicado se decidió separar el control del mismo de la EDU-CIAA por medio de opto acopladores. La tensión de alimentación de ésta última es de 5V y las salidas de los puertos proveen un nivel lógico alto de 3,3V. Se optó por establecer la tensión de alimentación externa en 15V de continua y utilizar sendos reguladores analógicos de 12V y 5V para proveer las tensiones necesarias.

Desde el punto de vista funcional, podemos dividir la placa en distintos sectores:

1) Display y teclado: se accede a los mismos a través de zócalos conectados a los puertos de entrada/ salida correspondientes[4].

2) Interfaz RS232: se utiliza un circuito HIN2320 para proveer los niveles necesarios para la comunicación a partir de los niveles LV-TTL que provee la EDU-CIAA.

3) Interfaz USB: Se utiliza la interfaz auxiliar de debug, a la que se accede a través de la UART0 de la EDU-CIAA.

4) Interfaz RS485: Se utiliza la interfaz provista por la EDU-CIAA.

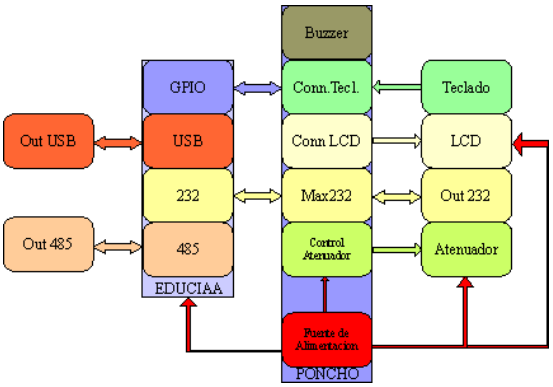

Fig. 1. Diagrama en bloques del sistema. Poncho integrado.

5) Salida hacia el atenuador [5]*:* se acondicionan las señales que comandan el atenuador, mediante el uso de optoacopladores, ya que tiene la ventaja de aislar completamente la EDU-CIAA del atenuador, evitando que corrientes no deseadas puedan dañar la placa controladora. Para compatibilizar la corriente que es capaz de entregar la EDU-CIAA en sus puertos de salida con la corriente necesaria para accionar las entradas del atenuador, se diseñó un circuito auxiliar que permite que un transistor operando como llave,

ISBN 978-987-46297-6-0

provea la corriente necesaria al optoacoplador sin cargar excesivamente las salidas de la EDU-CIAA. Dado que el dispositivo necesita 8 entradas que deben ser manejadas en forma independiente, se replicó el circuito otras tantas veces.

IV. Diseño del software

A los efectos del diseño de software se utilizó la biblioteca sAPI, desarrollada dentro del proyecto "Firmware_v2" [6]. Esta biblioteca nos permite tener una abstracción cómoda para el entendimiento y la programación de la placa y los periféricos utilizados.

Un aspecto importante que merece ser mencionado en el presente documento es el diseño, modelado, y elaboración del software implementado. El mismo fue desarrollado siguiendo una arquitectura *event-triggered* [7], que, generalmente implica el uso de múltiples interrupciones, cada una asociada a eventos periódicos específicos (como desborde de timers), o a eventos aperiódicos (como puede ser la entrada o salida de datos al sistema), lo que se ajusta perfectamente a éste caso en particular.

A su vez, se adoptó un modelo de máquinas de estado finitas; una principal, que maneja display, teclado y control del atenuador, y una secundaria, independiente de la primera, que se ocupa de configurar y manejar las interfaces de comunicación serie.

De esta manera, podría entenderse el software como constituido por capas (Fig. 2), encontrando así en la capa superior la máquina de estados principal que controla al atenuador, muestra el valor de atenuación en el display LCD y atiende las entradas de teclado. En una capa inferior, encontramos la segunda máquina de estados mencionada anteriormente, que se encarga de controlar el flujo de información con las interfaces de comunicación serie, esto es, tanto validar el comando recibido, como procesarlo adecuadamente y devolver la correspondiente respuesta; ésta es quien se vale de los drivers para comunicarse con los distintos periféricos involucrados en las tareas mencionadas anteriormente.

El lenguaje de programación utilizado para la codificación del algoritmo fue *C*, dentro del entorno de desarrollo *Eclipse*, en una de sus versiones adaptadas específicamente para el trabajo con la CIAA. Afortunadamente, contamos con un *Firmware* previamente desarrollado, que es de público acceso, y completamente gratuito, que nos proporciona facilidades a la hora de desarrollar el software dentro del entorno de programación.

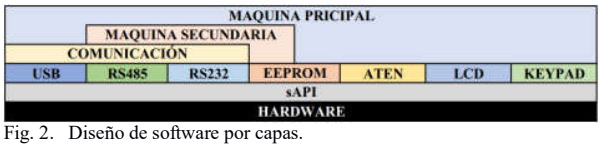

Fig. 2. Diseño de software por capas.

A la hora de la comunicación con las distintas interfaces serie, trabajando a modo *half-duplex*, decidimos seguir un formato estándar fijo, tanto para los comandos que envía el sistema como para los que recibe, con el objetivo de no solo facilitar el algoritmo, sino también para simplificar la posible comunicación con otros sistemas. Dicho formato, es el siguiente: "*nombreComando,parámetro.*"; en donde *nombreComando* hace referencia a qué parte del sistema se quiere configurar o conocer, mientras que en *parámetro* se encuentra un valor numérico o un operador que será de vital importancia para el intérprete de comandos. A continuación, algunos ejemplos de los posibles comandos: con "*ATEN,100.*" se establece el valor de la atenuación en 100 dB ; en cambio, con "*BR232,?.*" consultamos el *baudrate* al que está trabajando la interfaz serie RS232.

Por otra parte, las interfaces de comunicación serie poseen, a nivel de software, varios modos de trabajo, correspondientes a tres estados distintos: *PRIMARIA, SECUNDARIA, APAGADA*. Por aquella interfaz que se encuentre en el primero de ellos, el sistema será capaz tanto de enviar datos, como de recibir; si es, en cambio, *SECUNDARIA*, éste sólo podría recibirlos. Finalmente, si la interfaz se encuentra *APAGADA*, directamente se encuentra deshabilitada tanto la transmisión como la recepción de datos. Con respecto a las tasas de transmisión de cada interfaz, cabe aclarar que también son configurables por comando, o bien localmente por el teclado. Los parámetros de configuración, a su vez, son almacenados en memoria no volátil (EEPROM) para facilitar la operación del instrumento.

V. Resultados

Todos los ensayos realizados resultaron satisfactorios, a través de ellos se pudo validar implementación del hardware y del firmware de manera que el prototipo es funcional y se encuentra actualmente en uso (Fig. 3).

VI. Conclusiones

Como mejora posible del equipo, que actualmente está diseñado para uso en condiciones controladas de laboratorio, se podrían realizar algunas modificaciones, ya sea para su elaboración como producto o bien para su utilización portátil. En ese caso se podría reemplazar la EDU-CIAA por un hardware desarrollado en base a un procesador similar que permita la utilización de las mismas interfaces.

También para el caso de utilización como instrumento portátil, alimentado a baterías, podrían utilizarse algunas de las características de bajo consumo que provee el microcontrolador, para aumentar la duración de las mismas.

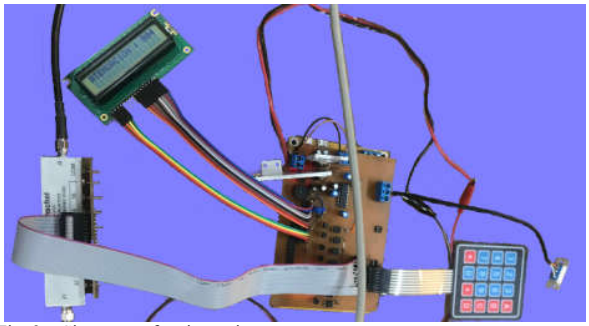
Fig. 3. Sistema en funcionamiento.

Referencias

[1] Modelos de Ponchos ya existentes, https://github.com/ciaa/Ponchos

[2] Manual del microcontrolador *lpc4337,* https://www.alldatasheet.es/datasheet-pdf/pdf/466540/NXP/LPC4337.html

[3] *Esquemáticos EDU-CIAA, http://www.proyecto-ciaa.com.ar/devwiki/doku.php*

[4] Mapeo de Puertos de Firmware, https://github.com/ciaa/firmware_v2/blob/master/sapi_examples/documentation/sAPI%20mapeo%20de%20perifericos%20EDU-CIAA-NXP.pdf

ISBN 978-987-46297-6-0

[5] *Manual del Atenuador Aeroflex/Weinschel 3200 series, web: https://www.apitech.com/products/rf-solutions/rf-components/attenuators/programmable-attenuator/electromechanical-relay-switched/3200-programmable-attenuator/*

[6] *Documentación del Firmware utiilizado, https://github.com/ciaa/firmware_v2/blob/master/sapi_examples/documentation/sAPI_reference.pdf*

[7] *Pont, M.J., "The Engineering of Reliable Embedded Systems: LPC 1769 edition", SafeTTy Systems. April 2015.*